**DAX**

**Why we need Dim Date Table or Calendar Table**

* To have a consistent and complete set of dates, even for days with no transactions or events.
* The Calendar Date table usually includes hierarchical structures such as Year, Quarter, Month, and Day. This hierarchy simplifies the process of different levels of granularity.
* Power BI provides Time Intelligence functions that work seamlessly with a Calendar Date table

Date and time functions

**GenerateSeries** - It will generate unique and incremental value.

GENERATESERIES ( Starting , Ending, incremental )

GENERATESERIES ( 1, 20, 1 )

**Randbetween -** It will generate random values between given value.

RANDBETWEEN(Bottom value, top value)

RANDBETWEEN(1, 10)

**DATE -** Returns the specified date in datetime format. If we have separate column like year, month, date by combining it we can get date.

DATE(<year>, <month>, <day>)

**DATEDIFF**

Returns the number of interval between two dates.

**DATEDIFF(<Date1>, <Date2>, <Interval>)**

date DIff = DATEDIFF(Orders[Order Date], Orders[Ship Date], YEAR )

date DIff = DATEDIFF(Orders[Order Date], Orders[Ship Date], DAY )

date DIff = DATEDIFF(Orders[Order Date], Orders[Ship Date], WEEK )

**example** :– order date – ship date = no of intervals

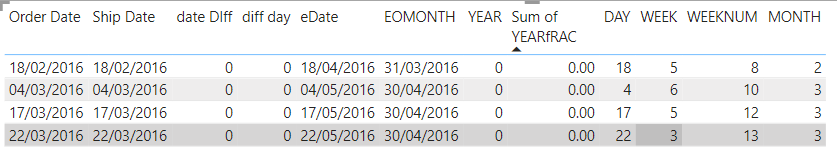
**Day, Month, Weekday, WeekNum, Year, Quarter**

WEEKNUM = WEEKNUM(Orders[Order Date])

DAY = DAY(Orders[Order Date]

MONTH = MONTH(Orders[Order Date])

YEAR = YEAR [Order Date])



# CALENDAR

# Returns a table with a range of dates is from the specified start date to the specified end date.

# if we enter end date 40,it will continue to next month.

# CALENDAR(<start\_date>, <end\_date>)

CALENDAR = CALENDAR(date(2019,08,17),date(2022,01,29))

# CALENDARAUTO

Returns a table the range of dates is calculated automatically based on data in the model.

**CALENDARAUTO(fiscal\_year\_end \_month)** whatever the values added in parenthesis it will increase that month in date column.

**Prerequisite is date column**

Min date **3-5-2020** and Max date **25-9-2023** CALENDAR AUTO generates date as per **1-1-2020 to Max date 31-12-2023**

# DATEVALUE

Converts a date in text format into a dateTime format.

Datevalue = DATEVALUE("02-09-2023")

**DATEDIFF**

Returns the number of interval between two dates.

**DATEDIFF(<Date1>, <Date2>, <Interval>)**

date DIff = DATEDIFF(Orders[Order Date], Orders[Ship Date], YEAR )

date DIff = DATEDIFF(Orders[Order Date], Orders[Ship Date], DAY )

date DIff = DATEDIFF(Orders[Order Date], Orders[Ship Date], WEEK )

**example** :– order date – ship date = no of intervals

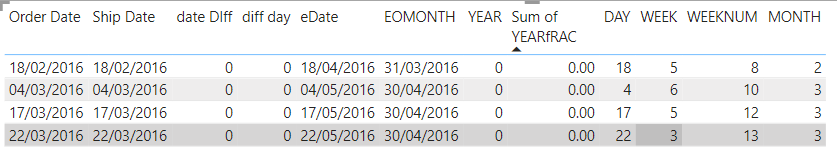
**Day, Month, Weekday, WeekNum, Year, Quarter**

WEEKNUM = WEEKNUM(Orders[Order Date])

DAY = DAY(Orders[Order Date]

MONTH = MONTH(Orders[Order Date])

YEAR = YEAR [Order Date])



**FORMAT**

FORMAT It will change the format of the date

FORMAT = FORMAT(DATEVALUE("02-09-2023"),"mm")

M - day 1

MM - day 01

MMM - JAN

MMMM - JANUARY

D - day 1

DD - Day 01

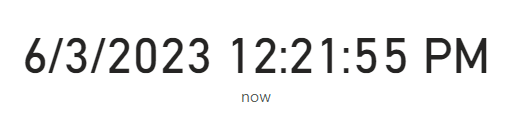
DDD - Thu

DDDD - Thursday

**NOW**

Returns the current date and time in **datetime** format.

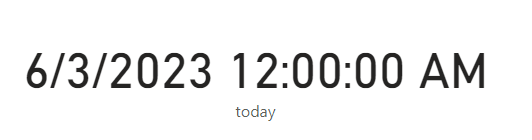
now = NOW()



**TODAY**

Returns the current date time is default as 12AM.

today = TODAY()

****

**UTC NOW**

Coordinated Universal Time (UTC). We have to add 5hr 30 min to get indian timings.

**TIMEVALUE**

Converts a time in text format to a time in datetime format.

**Syntax**

**TIMEVALUE(time\_text)**

timevalue = TIMEVALUE("17:26:23") 🡪🡪 5:26:23 PM

**hour, minute, second**

it will extract hour, minute, second from a given column

**Weeknum**

it will return no of week in a year like 52 weeks in a year.

**WeeKday**

it will return no of day in a week like 1 – 7 days.

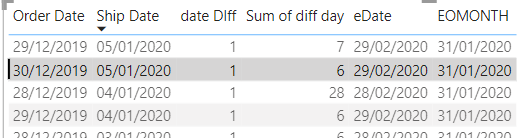
**EOMONTH**

Returns the date in date time format of the last day of the month

**EOMONTH(<start\_date>, <months>)**

First arg accept col and second arg accept no of month to be add or remove if I add 1 it will return next month last date

EOMONTH = EOMONTH([Order Date], 1 )



**EDATE**

Returns the dates that fall on the **same day of specified month**.

**EDATE(<start\_date>, <months>)**

eDate = EDATE(Orders[Order Date], 2)

Logical Function

If condition won’t accept the column name, use **selectedvalue,values or Distinct**

**Dax formatter in google to format Dax**

**IF**

Checks a condition, and returns one value when it's TRUE, otherwise it returns a second value.

sp = if( SELECTEDVALUE(sp[dynamic sp]) = "profit", SUM(Table1[Profit]), SUM(Table1[Sales]))

**SWITCH**

It is used for multiple condition

This function can be used to avoid having multiple nested [IF](https://learn.microsoft.com/en-us/dax/if-function-dax) statements.

region = SWITCH(Table1[Region], "west",1,"east",2,"north",3,"south",4)

reg = SWITCH(true(), Table1[Region]="west", 1, Table1[Region]="east" , 2, Table1[Region]="north",3,Table1[Region]="south",4)

**AND**

If all conditions are true then only returns true otherwise false.

week = if('date table'[day]>=1 && 'date table'[day]<=7,1, BLANK())

week = if( AND ( 'date table'[day]>=1 , 'date table'[day]<=7),1, BLANK())

**OR**

If any one conditions are true then only returns true otherwise false.

week = if('date table'[day]>=1 || 'date table'[day]<=7,1, BLANK())

week = IF( OR ( 'date table'[day]>=1 , 'date table'[day]<=7),1, BLANK())

Time intelligence functions

Calculations over those periods.

**TOTALMTD, TOTALQTD, TOTALYTD**

To calculate cumulative of current month,current year,current quarter.

cumulative = TOTALMTD( [sumsales], Table1[Order Date])

cumulative = TOTALQTD( [sumsales], Table1[Order Date])

cumulative = TOTALYTD( [sumsales], Table1[Order Date])

**DATESMTD, DATESQTD, DATESYTD**

* Returns a table that contains a column of the dates for the month,quarter,year to date, in the current context.
* It will take max Month, Year, Quarter in the model.
* It won’t work separately, is should works in calculate function, for cumulative operation.

**Syntax**

**DATESMTD(<dates>)**

curent QTD = TOTALQTD([TOtal sales], DATESQTD(Orders[Order Date]))

current MTD = TOTALMTD([TOtal sales],DATESMTD(Orders[Order Date]))

current YTD = TOTALYTD([TOtal sales],DATESYTD(Orders[Order Date]))

previous MTD = CALCULATE([TOtal sales], PREVIOUSMONTH (DATESMTD(Orders[Order Date])))

Or

previous MTD = TOTALMTD([TOtal sales], PREVIOUSMONTH (DATESMTD(Orders[Order Date])))

previous QTDs = CALCULATE([TOtal sales], PREVIOUSQUARTER (DATESQTD(Orders[Order Date])))

SPLY = CALCULATE( [current MTD], SAMEPERIODLASTYEAR(Orders[Order Date]))

**OPENINGBALANCEMONTH**

Evaluates the **expression** at the first date of the month in the current context.

OBM = OPENINGBALANCEMONTH([TOtal sales],Orders[Order Date])

OQM = OPENINGBALANCEQUARTER([TOtal sales],Orders[Order Date])

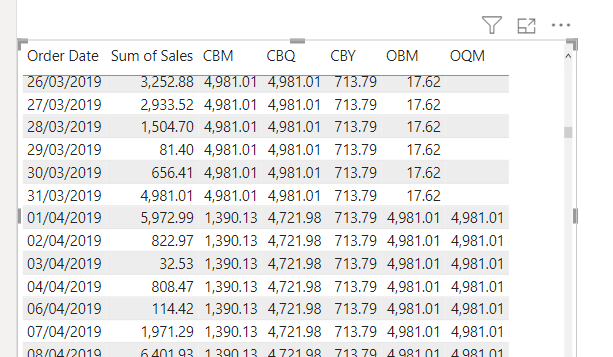
**CLOSINGBALANCEMONTH**

Evaluates the **expression** at the last date of the month in the current context.

CBM = CLOSINGBALANCEMONTH([TOtal sales],Orders[Order Date])

CBQ = CLOSINGBALANCEQUARTER([TOtal sales],Orders[Order Date])

CBY = CLOSINGBALANCEYEAR([TOtal sales], Orders[Order Date])



**ClosingBalanceMonth** will give the last day of month with some expression.

**EOMONTH** will give only the date of Last day of month.

**DATESBETWEEN**

Returns a table that contains a column of dates that begins with a specified start date and continues until a specified end date.

DATESBETWEEN(<Dates>, <StartDate>, <EndDate>)

DatesBetweeen = CALCULATE([TOtal sales], DATESBETWEEN(Orders[Order Date],DATE(2019,1,1),DATE(2019,6,31)))

**DATESINPERIOD**

Returns a table that contains a **column of dates** that begins with a specified **start date** and continues for the specified number and type of date intervals.

Till the interval time it will sum values, provides Cumulative values

DATESINPERIOD(<dates>, <start\_date>, <number\_of\_intervals>,<interval>)

Dates in period = CALCULATE([TOtal sales], DATESINPERIOD( Orders[Order Date].[Date], DATE(2017,1,1), -1,YEAR))

DIP = CALCULATE([TOtal sales], DATESINPERIOD(Orders[Order Date].[Date], DATE(2018,6,12), 2,QUARTER))

**SAMEPERIODLASTYEAR**

Returns a table that contains a column of dates shifted one year back in time from the dates in the specified **dates** column, in the current context.

SAMEPERIODLASTYEAR(<dates>)

current QTD = TOTALQTD([TOtal sales], DATESQTD(Orders[Order Date]))

SPLY MTD = CALCULATE([current MTD], SAMEPERIODLASTYEAR(Orders[Order Date]))

SPLY QTD = CALCULATE([curent QTD],SAMEPERIODLASTYEAR(Orders[Order Date]))

**RankX**

Assign the rank value for each row

r\_rank = RANKX( ALL(Table1[Product ID]) ,[sumsales],,DESC)

for rankx ALL() filter is mandatory and measures we have to do instead of direct calculation.

**PARALLELPERIOD**

Returns a table that contains a column of dates that represents a period parallel to the dates in the specified **dates** column, in the current context, with the dates shifted a number of intervals either forward in time or back in time.

**parallel period** will works as **samePeriod last year** but its is limited to single year, more than 1 year we can use ParallelPeriod.

PARALLELPERIOD(<dates>,<number\_of\_intervals>,<interval>)

parallel period = CALCULATE([mtd],PARALLELPERIOD(Orders[Order Date].[Date],-2,YEAR))

**DATEADD**

Returns a table that contains a column of dates, shifted either forward or backward in time by the specified number of intervals from the dates in the current context.

DATEADD(<dates>,<number\_of\_intervals>,<interval>)

**DATEADD** will works as **parallel period** but it is limited to month,year,quarter. But in dateadd it has month,year,quarter,day.

**examples**

**[** TOtalYTD = TOTALYTD(SUM(Orders[Sales]), date\_auto[Date])

**it will return only previous 2nd year result**

dateADD = CALCULATE( [TOTALYTD CAL], DATEADD(date\_auto[Date],-2,YEAR)) **]**

**it will return from starting to till previous 2 year result only.**

**[** dateADD = CALCULATE(SUM(Orders[Sales]),], DATEADD(date\_auto[Date],-2,YEAR)) **]**

**Running Total or cumulative value**

Running = CALCULATE([Sum\_sales], DATESYTD(Dim\_date[Date]))

**Running total till -20 days of record**

Sum\_sales = SUM(Orders[Sales])

DateAdd = CALCULATE([Sum\_sales], DATEADD(Dim\_date[Date],-20,DAY))

Running = CALCULATE([DateAdd], DATESYTD(Dim\_date[Date]))

Note - if we use **dim date** values will behave based on **dimDate** or if we use **order date** data will behave on **order date**.

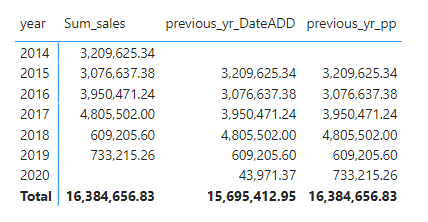
If we use dim date column in table, all column of dax should be on dim date only it wont work.

**Sales compare to previous year**

a = [Sum\_sales] - [DateAdd]

In **Date Add** if we have **10 months** in current months if give mention **-1 year** it will calculate **10 months** of previous months only.

In Parallel period if we have **10 months** in current months if give mention -**1 year** it will calculate **whole year** of previous months only.



**PREVIOUSDAY**

prevoius = CALCULATE(SUM(Orders[Sales]), PREVIOUSDAY(Orders[Order Date]))

for previous month we have to select date of fact table not DIM Date table

**NextDay**

NextDay = CALCULATE(SUM(Orders[Sales]), NEXTDAY(Orders[Order Date]))

**FirstDate**

MIN = FIRSTDATE(Orders[Order Date])

MIN = MIN(Orders[Order Date])

It will return min date in the model

**LastDate**

LastDate = LASTDATE( Orders[Order Date])

MIN = MAX(Orders[Order Date])

It will return Max date in the model